UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/678,145	10/03/2000	Stuart John Macdonald	002114.P012	6983

7590 03/26/2004

Sheryl Sue Holloway
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 Wilshire Boulevard 7th Floor
Los Angeles, CA 90025

EXAMINER
BRANCOLINI, JOHN R

ART UNIT	PAPER NUMBER
2153	2

DATE MAILED: 03/26/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PPG

Office Action Summary	**Application N** 09/678,145	**Applicant(s)** MACDONALD ET AL.	
	Examiner John R Brancolini	**Art Unit** 2153	

-- The MAILING DATE of this communication app ars on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 03 October 2000.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-30 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-30 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on 03 October 2000 is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

Claims 1-30 are pending in the application.

Priority

No claim for priority has been made. The effective filing date is October 3, 2000.

Drawings

Figures 1 and 2 should be designated by a legend such as --Prior Art-- because only that which is old is illustrated. See MPEP § 608.02(g). A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description:

- Page 16 line 19 states an arrow number 587. In figure 5A, the arrow is numbered 537.

A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Specification

The disclosure is objected to because of the following informalities:

- Page 16, line 11, line 18, line 19 the circuit flow object is referred to as number 537, while on Figure 5A the circuit flow object item is numbered 535.

Appropriate correction is required.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1,2 6-12, 16-19, 22-25, 29-30 are rejected under 35 U.S.C. 102(e) as being anticipated by Bennett et al. (US Patent Number 6122670), hereinafter referred to as Bennett.

In regards to claim 1, Bennett discloses a computerized method for sequencing and reassembling messages from protocol data units exchanged in a communications channel between two computers, the method comprising:

- Creating a protocol flow object to represent each protocol layer used by the communications channel, each protocol flow object having a circuit element associated with each transmission direction in the channel (In Figures 2a and 2b, Bennett shows a functional block diagram of the system where each protocol layer, such as the transport layer, the network layer, and the datalink layer, are represented by individual flow objects, each with bi-directional communication channels, see also col 4 line 10 to col 5 line 4, a brief overview of the system including details on each protocol layer).

- Arranging the protocol flow objects in a logical tree structure that mirrors a hierarchy for the protocol layers (Figures 2a and 2b show a block diagram of the Bennett system, which is built in a logical tree structure mirroring a hierarchy where each protocol layer flows into the next).
- Creating circuit flow objects for each protocol layer to represent the protocol data units for the protocol layer immediately higher in the hierarchy (Bennett shows that each layer creates a new flow object based on data sets buffered from the protocol layer directly above the current layer in the hierarchy, see figures 2a and 2b, col 4 lines 10-29).
- Associating a transmission direction with each circuit flow object (after being handled by a protocol layer, the flow object is given a transmission direction to continue through the hierarchy, col 4 lines 50-53, instructions are included with the data as to which layer it is to be forwarded to).
- Linking each circuit flow object for a protocol layer to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object (after being handled by a protocol layer, the

flow object is given a transmission direction to continue through the hierarchy, col 4 lines 50-53, instructions are included with the data as to which layer it is to be forwarded to, first the data is buffered then it is linked to the correct protocol layer circuit element).

- Sequencing the circuit flow objects linked to a particular protocol flow object when specified by the protocol layer represented by the particular protocol flow object (individual flow objects are initially buffered when specified for a particular protocol layer which creates a method of sequencing, see col 4 lines 50-53, 61-64).
- Reassembling the messages from the circuit flow objects linked to the protocol flow object at the top of the tree structure (ATM network interface chip performs packet reassembly at the logical top of the tree structure, col 6 lines 23-31).

In regards to claim 2, Bennett discloses creating the circuit flow objects for each protocol layer comprises:

- Creating the circuit flow objects for the protocol flow object at the bottom of the tree structure by extracting data from the protocol data units for the protocol layer lowest in the hierarchy (each level of the tree is created from extracting the buffered data supplied by the next lowest protocol layer, this can be seen by the TCP process in the transport layer loading the data from the buffer supplied by the FTP process, which is logically lower in the hierarchy, col 4 lines 45-53).

- Creating the circuit flow objects for the remaining protocol flow objects in the tree structure by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure (as shown above, each level of the tree is created from extracting the buffered data supplied by the next lowest protocol layer, this can be seen by the IP process in the network layer loading the data from the buffer supplied by the TCP process, which is logically lower in the hierarchy, col 4 lines 60-68).

In regards to claim 6, Bennett discloses the protocol flow objects are created in order from the bottom to the top of the hierarchy (Bennett describes the creation flow of the protocol objects in detail in col 4 line 10 – col 5 line 4, logically the creation of the objects is in order from the bottom to the top of the hierarchy).

In regards to claim 7, Bennett discloses the circuit flow objects for a current protocol flow object are created before creating the protocol flow object for the protocol layer immediately above the current protocol flow object in the hierarchy (each level is executed one at a time, before the data is buffered and sent to the next level for a new circuit flow execution, one example is the execution and linking of the data at the TCP process level, col 4 lines 45-53).

In regards to claim 8, Bennett discloses arranging the protocol flow objects into a logical tree structure comprises:

- Creating multiple branches in the tree structure when a plurality of protocol layers are immediately above a current protocol layer in the hierarchy, each branch corresponding to one of the plurality of protocol layers (Figure 2A shows multiple branches between the transport layer and the network layer, each branch either corresponding to the UDP process or the TCP process).

In regards to claim 9, Bennett discloses determining the protocol layers in the hierarchy (each layer is determined based on linking information and buffering location, col 4 lines 10-29).

In regards to claim 10, Bennett discloses storing the protocol flow objects and the circuit flow objects in a flow object database (Figure 14B shows the storing of the separate flow objects into a data base in the hard disk 25).

In regards to claim 11, Bennett discloses a computer-readable medium having computer-executable instructions to a cause a computer to perform a method comprising:

- Creating a protocol flow object to represent each protocol layer used by a communications channel, each protocol flow object having a circuit element associated with a transmission direction in the channel (In Figures 2a and 2b, Bennett shows a functional block diagram of the system where each protocol layer, such as the transport layer, the network layer, and the data link layer, are represented by individual flow objects, each with bi-directional communication channels, see also col 4 line 10 to col 5 line 4, a brief overview of the system including details on each protocol layer).

- Arranging the protocol flow objects in a logical tree structure that mirrors a hierarchy for the protocol layers (Figures 2a and 2b show a block diagram of the Bennett system, which is built in a logical tree structure mirroring a hierarchy where each protocol layer flows into the next).
- Creating circuit flow objects for each protocol layer to represent the protocol data units for the protocol layer immediately higher in the hierarchy (Bennett shows that each layer creates a new flow object based on instruction sets received from the higher level in the hierarchy, col 4 lines 10-29).
- Associating a transmission direction with each circuit flow object (after being handled by a protocol layer, the flow object is given a transmission direction to continue through the hierarchy, col 4 lines 50-53, instructions are included with the data as to which layer it is to be forwarded to).
- Linking each circuit flow object for a protocol layer to the circuit element of the representative protocol flow object that matches the transmission direction associated with the circuit flow object (after being handled by a protocol layer, the flow object is given a transmission direction to continue through the hierarchy, col 4 lines 50-53, instructions are included with the data as to which layer it is to be forwarded to, first the data is buffered then it is linked to the correct protocol layer circuit element).

- Sequencing the circuit flow objects linked to a particular protocol flow object when specified by the protocol layer represented by the particular protocol flow object (individual flow objects are initially buffered when specified for a particular protocol layer which creates a method of sequencing, see col 4 lines 50-53, 61-64).
- Reassembling the messages from the circuit flow objects linked to the protocol flow object at the top of the tree structure (ATM network interface chip performs packet reassembly at the logical top of the tree structure, col 6 lines 23-31)..

In regards to claim 12, Bennett discloses computer-executable instructions comprising:

- Creating the circuit flow object for the protocol flow object at the bottom of the tree structure by extracting data from the protocol data units for the protocol layer lowest in the hierarchy (each level of the tree is created from extracting the buffered data supplied by the next lowest protocol layer, this can be seen by the TCP process in the transport layer loading the data from the buffer supplied by the FTP process, which is logically lower in the hierarchy, col 4 lines 45-53).
- Creating the circuit flow objects for the remaining protocol layers by extracting data from the circuit flow objects linked to the protocol flow object immediately lower in the tree structure (as shown above, each level of the tree is created from extracting the buffered data supplied by the next lowest protocol layer, this can be seen by the IP process in the network layer loading the data from the buffer

supplied by the TCP process, which is logically lower in the hierarchy, col 4 lines 60-68).

In regards to claim 16, Bennett discloses computer-executable instructions comprising:

- Creating multiple branches in the tree structure when a plurality of protocol layers are immediately above a current protocol layer in the hierarchy, each branch corresponding to one of the plurality of protocol layers (Figure 2A shows multiple branches between the transport layer and the network layer, each branch either corresponding to the UDP process or the TCP process).

In regards to claim 17, Bennett discloses determining the protocol layers in the hierarchy (each layer is determined based on linking information and buffering location, col 4 lines 10-29).

In regards to claim 18, Bennett discloses storing the protocol flow objects and the circuit flow objects in a flow object database (Figure 14B shows the storing of the separate flow objects into a data base in the hard disk 25).

In regards to claim 19, Bennett discloses a computer-readable medium having stored thereon a protocol flow object data structure comprising:

- A key field containing data representing an identifier for a connection between two computers at a protocol layer (fig 5 item 228 shows an ATM cell routing data which contains data representing the connection path for two computers).
- A primary circuit element containing data representing a link to a series of protocol data units flowing in one direction in the connection identified by the key field (an element is present which is used by each process, such as the TCP process, for link data in one direction, such as for transmitting data, col 4 lines 45-53).
- An alternate circuit element containing data representing a link to a series of protocol data units flowing in an opposite direction in the connection identified by the key field (an alternate element is present which is used by each process, such as the TCP process, for link data in a second direction, such as for receiving data, col 4 lines 45-53).

In regards to claim 22, Bennett discloses a computer-readable medium having stored thereon a flow object data structure comprising:

- A plurality of protocol flow objects, each protocol flow object comprising:
    - A key field containing data representing an identifier for a connection between two computers at a protocol layer (fig 5 item 228 shows an ATM cell routing data which contains data representing the connection path for two computers).

  - A primary circuit element containing data representing a link to a series of protocol data units flowing in one direction in the connection identified by the key field (an element is present which is used by each process, such as the TCP process, for link data in one direction, such as for transmitting data, col 4 lines 45-53).
  - An alternate circuit element containing data representing a link to a series of protocol data units flowing in an opposite direction in the connection identified by the key field (an alternate element is present which is used by each process, such as the TCP process, for link data in a second direction, such as for receiving data, col 4 lines 45-53).
- A tree structure comprising a plurality of entries, each entry comprising:
  - A protocol field containing data representing the identifier for one of the plurality of protocol flow objects (Figure 5 shows a packet of data containing several protocol fields).
  - A lower protocol field containing data representing the identifier for the protocol flow object immediately lower in a protocol layer hierarchy relative to the protocol flow object identified by the protocol field (a second data field is present in the data packet shown in figure 5, here represented by the TCP header, see also col 6 lines 44-56).
  - A higher protocol field containing data representing the identifier for the protocol flow object immediately higher in the protocol layer hierarchy relative to the protocol flow object identified by the protocol field (a first

data field is present in the data packet shown in figure 5, here represented by the IP header, see also col 6 lines 44-56)..

In regards to claim 23, Bennett discloses a computer-readable medium further comprising:

- A plurality of circuit flow objects, each circuit flow object containing data representing one of the protocol data units (Bennett shows that each layer creates a new flow object based on instruction sets received from the higher level in the hierarchy, col 4 lines 10-29).

In regards to claim 24, Bennett discloses a computerized system comprising:

- A processor (figure 3 item 10 shows a processor).
- A memory coupled to the processor through a bus (figure 3 item 15 shows memory coupled to the processor through a bus).
- A computer-readable medium coupled to the processor through the bus (figure 3 item 25 shows a computer hard disk).
- A plurality of protocol interpreters stored on the computer-readable medium for execution by the processor (the hard disk contains processes for use by the operating system, including protocol interpreters, col 5 lines 44-59).
- A decode engine executed from the computer-readable medium to cause the processor to

- Create protocol flow objects representing protocol layers and circuit flow objects representing data flows at the protocol layers (Bennett shows that each layer creates a new flow object based on instruction sets received from the higher level in the hierarchy, col 4 lines 10-29).
- Extract data from the circuit flow objects representing protocol data units at a particular protocol layer as directed by one of the protocol interpreters (data is extracted and sets of instructions executed at each layer, col 4 lines 17-29).
- Sequence the circuit flow objects representing the protocol data units at a particular protocol layer if directed by one of the protocol interpreters (individual flow objects are initially buffered when specified for a particular protocol layer which creates a method of sequencing, see col 4 lines 50-53, 61-64).
- Reassemble messages from the circuit flow objects representing the protocol data units at a particular protocol layer if directed by one of the protocol interpreters (ATM network interface chip performs packet reassembly at the logical top of the tree structure, col 6 lines 23-31).

In regards to claim 25, Bennett discloses the decode engine further causes the processor to store the protocol flow objects and circuit flow objects in a flow database (Figure 14B shows the storing of the separate flow objects into a data base in the hard disk 25), logically link the protocol flow objects into a hierarchical tree structure (Figures

2a and 2b show a block diagram of the Bennett system, which is built in a logical tree structure mirroring a hierarchy where each protocol layer flows into the next), and to logically link the circuit flow objects to the protocol flow objects (after being handled by a protocol layer, the flow object is given a transmission direction to continue through the hierarchy, col 4 lines 50-53, instructions are included with the data as to which layer it is to be forwarded to, first the data is buffered then it is linked to the correct protocol layer circuit element).

In regards to claim 29, Bennett discloses a method of communicating between a protocol interpreter and a segmentation and re-assembly decode engine for a communications network comprising:

- Issuing, by the protocol interpreter, an add data unit command (the first protocol process, the FTP process, computes an add unit command to initialize a new transfer, col 4 lines 10-29, also figures 2A and 2B).
- Receiving, by the segmentation and re-assembly decode engine, the add data unit command (both the TCP protocol and IP protocol receive the data unit commands, for both segmentation and re-assembly of the message, col 4 lines 2-9, 45-67).
- Issuing, by the segmentation and re-assembly decode engine in response to receiving the add data unit command, an instruction to a flow object data base (both the TCP protocol and IP protocol continue to forward the data unit commands, for both segmentation and re-assembly, col 4 lines 45-67).

In regards to claim 30, Bennett discloses the instruction is selected from the group consisting of adding a circuit flow object to the flow object data base (the instruction is fetched by the CPU from a predetermined instruction set, col 4 lines 17-20), associating a circuit flow object to a protocol flow object in the flow object data base (Figure 14B shows the storing of the separate flow objects into a data base in the hard disk 25), retrieving a circuit flow object from the flow object data base (figures 2A and 2B show the FTP process retrieving flow objects from the data stored in hard disk 25), and sequencing a circuit flow object relative to other circuit flow objects in the data base (individual flow objects are initially buffered when specified for a particular protocol layer which creates a method of sequencing, see col 4 lines 50-53, 61-64).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 3-5, 13-15, 21, 26-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bennett in view of Stevens (TCP/IP Illustrated, Volume 1; Stevens, W Richard; Addison Wesley Publishers, 1994; pages 148-151).

In regards to claims 3, 13 and 26, Bennett discloses all the limitations of the independent claims 1, 11, and 24, but fails to disclose a vector list to represent fragmented data. Bennett discusses the use of the IP protocol process, but fails to directly disclose the IP process creation of a vector to maintain fragmentation information.

Stevens presents a detailed explanation of IP fragmentation of data, as well as detailing how an IP process creates a vector listing to represent the fragmented data for reassembly. On pages 148-151 Stevens discusses IP fragmentation, and in detail on page 149 Stevens shows how a listing is created stored in an identification field to represent fragmented data. This re-assembly vector allows a message to be re-assembled for receiving and re-fragmented for retransmission quickly and efficiently.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify Bennett to include a vector list to represent fragmented data as taught by Stevens to allow a message to be re-assembled for receiving and re-fragmented for retransmission quickly and efficiently.

In regards to claims 4, 14, and 27, Stevens discloses a vector list comprises a vector specifying a protocol data unit number (page 149, paragraph 2 says an identification field is used to specify a data unit number), a length value (page 149, paragraph 2, a total length field is present), and an offset value for each fragment of the fragmented data (page 149, paragraph 2, a fragment offset field is provided).

In regards to claims 5, 15, and 28, Stevens discloses reassembling the fragmented data in accordance with the vectors in a vector list (page 148, last paragraph discusses reassembling the message at the receiver based on the vector list information).

In regards to claim 21, Bennett fails to disclose a re-assembly vector. Stevens however, discloses a re-assembly vector comprising:

- A protocol data unit field containing data representing a number for a protocol data unit (page 149, paragraph 2 says an identification field is used to specify a data unit number).
- A length field containing data representing a length of a data payload in the protocol data unit identified by the protocol data unit field (page 149, paragraph 2, a total length field is present).
- An offset field containing data representing a starting position of the data payload in the protocol data unit identified by the protocol data unit field (page 149, paragraph 2, a fragment offset field is provided).

This re-assembly vector allows a message to be re-assembled for receiving and re-fragmented for retransmission quickly and efficiently.

It would have been obvious to one of ordinary skill in the art at the time of invention to modify Bennett to include a re-assembly vector as taught by Stevens to allow a message to be re-assembled for receiving and re-fragmented for retransmission quickly and efficiently.

Claim Rejections - 35 USC § 103

Claim 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over Bennett in view of Schoffelman et al. (US Patent Number 6119170), hereinafter referred to as Schoffelman.

In regards to claim 20, Bennett discloses all the limitations of claim 19, but fails to disclose the links comprise hash tables for identifying the series of data units.

Schoffelman discloses a system of transporting messages on a TCP/IP network to a different TCP/IP network with support for a variety of support for different application types. Figure 3a shows a table of links formed into a hash table for directing data entries (see also col 6 lines 41-63). Schoffelman teaches that using a hash table to store links allows fast access to a listing of links.

It would have been obvious at the time of invention to modify Bennett to include a hash table for identifying series of data units comprised of a set of links as taught by Schoffelman to allow fast access to a listing of links.

Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure:

- Gulick et al (US Patent 6014709), a system for controlling a flow of messages by constructing a hierarchal tree of caches and vector files to maintain the message fragments in each logical level.

- Hansen (US Patent 6697871), a system and method for encoding and decoding protocol messages.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to John R Brancolini whose telephone number is (703) 305-7107. The examiner can normally be reached on M-Th 7am-5:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Glenton Burgess can be reached on (703) 305-4792. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

JRB

FRANTZ B. JEAN
PRIMARY EXAMINER